10. ročník Žilina 2. – 4. 7. 2018
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
OSS Conf
ossconf.soit.sk
OSSConf 2025: 1–10

Vítejte ve světě animací!

PAVEL STŘÍŽ (CZ)

Abstrakt. Článek představuje základní možnosti animování grafiky ve světě TEXu.
Klíčová slova. dvisvgm, svganimation, media4svg.

WELCOME TO THE WORLD OF ANIMATIONS!

Abstract. The article introduces basic options of animating graphics in the TEX world.
Keywords. dvisvgm, svganimation, media4svg.

1. Vstup do světa xml

Formát xml jako rozšíření html asi netřeba blíž představovat. Vyřešil starší problém strukturování dat nad rámec dat v tabulce řádky krát sloupce na straně jedné a relačních databází na straně druhé. S xml se potkáváme u MathML a především ConTEXt tomu věnoval velkou pozornost.

`http://pragma-ade.com/show-man-7.htm`

`http://pragma-ade.com/general/manuals/xml-mkiv.pdf`

Zájemce o tuto problematiku odkazuji na knihu Dana Lynche z roku 2020 *The Art of Digital Publishing*, `https://mathapedia.com/books/31`, kapitolu 6: The Mathematical Web.

U grafiky přichází formát svg, textový formát pracující v mezích xml. Především program Inkscape zaznamenal velkou oblibu ve světě open source, svg používá jako nativní formát s možností importu a exportu do pdf, včetně možnosti přes příkazový řádek a parametr `--export-pdf`. Ve světě TEXu byla grafika vždy trochu pozadu a plní trochu jiné úkoly než na které jsou grafici a animátoři zvyklí. TikZ umí načíst svg. Jisté usnadnění dávají balíčky svg, svg-extract, starší balíček svg-inkscape a tikztosvg, v době psaní tohoto článku ještě nebyl zařazen do TEXLive.

```
$ texdoc svg svg-extract svg-inkscape
$ firefox https://ctan.org/pkg/tikztosvg
```

Nyní se nám podaří otevřít pdf přes Inkscape, nabídne nám možnosti přes knihovnu Poppler/Cairo či přes upravenou variantu knihovny Poppler. Pokud navolíme Internal import a odškrtneme Replace PDF fonts by closest-named installed fonts, dá se s obrázkem pracovat, byť texty se nedají editovat, jsou z nich křivky.

Jaromír Antoch tuto cestu podrobněji zkoušel a u některých starších příspěvků se text jakoby rozsype. Dávám to za vinu starším písmům ještě v rastrovém formátu. Asi by si to zasloužilo ještě bádání.

TikZ umí vygenerovat svg, viz kapitola 10.2.4 v manuálu verze 3.1.5b, závisí však na nástroji `dvisvgm`.

```
$ texdoc tikz
```

2. dvisvgm v2.9.1

Na následující testy jsem si připravil zatěžkávací dokument, pracovně soubor `100-pisma.tex`. Znaky s diakritikou, rastrové emodži a kousek japonské básně jako zástupce jazyků ČJKV.

```
\documentclass{article}
\usepackage{emoji}
\usepackage{luatexja}
\begin{document}\pagestyle{empty}
\huge\noindent
Ó, náhlý déšť již zvířil prach a čilá laň teď běží s houfcem gazel
k úkrytům.\emoji{baby}\emoji{sparkling-heart}\emoji{speak-no-evil-monkey}
鳥啼く声す 夢覚ませ 見よ明け渡る 東を 空色栄えて 沖つ辺に 帆船群れゐぬ 靄の中
\end{document}
```

Jeden ze starších pokusů jak získat svg je nástroj `pdf2svg`. To bude pro mne srovnávací dokument.

```
$ sudo apt install pdf2svg
```

Spouštíme a dostáváme první obrázek ze čtyř dále v textu.

```
$ lualatex 100-pisma.tex
$ pdf2svg 100-pisma.pdf 100-pisma-pdf2svg.svg
```

Nástroj `dvisvgm` má domovskou stránku `https://dvisvgm.de`.

```
$ man dvisvgm
$ info dvisvgm
```

U písem si musíme dát pozor a případně zvolit přepínač `-n` (bez zařazení písem). Zde je ukázka rozdílu při aplikaci na ukázkový dokument z `http://ctan.math.illinois.edu/macros/latex/contrib/media4svg/example/`.

```
$ dvilualatex beamer-example.tex
$ dvilualatex beamer-example.tex
$ dvisvgm --bbox=papersize --font-format=woff2 --zoom=-1 --page=-
    beamer-example.dvi
#$ dvisvgm -n --bbox=papersize --font-format=woff2 --zoom=-1 --page=-
    beamer-example.dvi
```

Rozdíl mezi 3. a případným 4. příkazem je viditelný. Došlo k náhradě písem a umístění glyfů nesedí. I kdyby vše sedělo, zdrojový kód svg je prakticky ručně needitovatelný.

Player control

The standardplayer controls (option 'controls ') take a lot of spaceof the media display. Thereforeit is not recommendedto enablethem. Nevertheless, interactivity is still providedthroughtouch or left mousebuttonclick, and through the keyboardas summarizedin the table.
Click on the mediadisplay to start playback. To pauseplayback, pressthe left mousebutton on the mediadisplay. Releaseit to resumeplayback. To pause playback permanently, pressthe left mousebutton on the mediadisplay and move the mouseout whilekeepingthe button pressed.

Command	Shortcut	Command	Shortcut
TogglePlay/P ause	Space	Seekback 1 %	←
Increasevolume	↑	Seekforward 1 %	→
Decreasevolume	↓	Seekback 10%	Ctrl + ←
Unmute audio	Ctrl + ↑	Seekforward 10%	Ctrl + →
Mute audio	Ctrl + ↓	Seekto beginning	Home
ToggleFull-Screen	F11	Seekto end	End

Player control

The standard player controls (option 'controls') take a lot of space of the media display. Therefore, it is not recommended to enable them. Nevertheless, interactivity is still provided through touch or left mouse button click, and through the keyboard as summarized in the table.
Click on the media display to start playback. To pause playback, press the left mouse button on the media display. Release it to resume playback. To pause playback permanently, press the left mouse button on the media display and move the mouse out while keeping the button pressed.

Command	Shortcut	Command	Shortcut
Toggle Play/Pause	Space	Seek back 1 %	←
Increase volume	↑	Seek forward 1 %	→
Decrease volume	↓	Seek back 10 %	Ctrl + ←
Unmute audio	Ctrl + ↑	Seek forward 10 %	Ctrl + →
Mute audio	Ctrl + ↓	Seek to beginning	Home
Toggle Full-Screen	F11	Seek to end	End

Kdyby nástroj nemohl dohledat písma GhostScriptu, užívá se k tomu parametr `--libgs`. Pokud užijeme náhradu písma, je dobré zvolit parametr `-e` na přesný výpočet bounding boxu glyfů. Je to podobné jako u nástroje `pdfcrop`. Můžeme zvolit cestu tex→dvi/xdv→svg, ale i tex→pdf/ps→svg.

Dokumentace doporučuje užít další parametry: `--font-format=woff` nebo `woff2` na nastavení formátu písma, `--zoom=-1` aby se nezasahovalo do velikosti stran, `--page=1,-` pro volbu všech stran, jinak se bere jen první strana, `--optimize` pro optimalizaci výsledného svg, případně i `-z` / `--precision=1` na nastavení počtu desetinných míst. Na zobrazení svg doporučují prohlížeče Chrome, Chromium a Opera, Firefox se zdá být pomalejší.

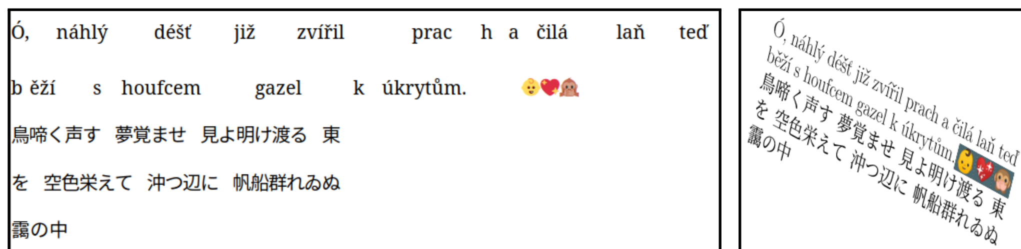
U našeho zatěžkávacího dokumentu spouštíme:

```
$ dvilualatex 100-pisma.tex
$ dvisvgm -n --zoom=-1 --page=- -o 100-pisma-prespdf1.svg --pdf 100-pisma.pdf
$ dvisvgm --font-format=woff2 --exact --zoom=-1 --page=- -o 100-pisma-spismy.svg
    100-pisma.dvi
$ dvisvgm -n --zoom=-1 --page=- -o 100-pisma-prespdf2.svg --pdf
    --transform="R20,w/3,2h/5 T1cm,1cm S2,3" 100-pisma.pdf
```

Níž jsou náhledy. První řádek nám vysází dokument do dvi (1. obrázek i identický výsledek nástrojem `pdf2svg`), druhý převodem písem do vektorových křivek, třetí se pokusí o vysázení s náhradou písem a poslední příkaz je ukázka geometrické transformace celé stránky. Je vidět, že nástroj má nějakou závadu u pozadí rastrových emodži, jinak je výsledek uspokojivý při převodu písem do křivek.

Ó, náhlý déšť již zvířil prach a čilá laň teď běží s houfcem gazel k úkrytům. 👶💖🙈
鳥啼く声す 夢覚ませ 見よ明け渡る 東を 空色栄えて 沖つ辺に 帆船群れゐぬ 靄の中

Ó, náhlý déšť již zvířil prach a čilá laň teď běží s houfcem gazel k úkrytům. 👶💖🙈
鳥啼く声す 夢覚ませ 見よ明け渡る 東を 空色栄えて 沖つ辺に 帆船群れゐぬ 靄の中



3. animate + `dvisvgm`

Ve světě JavaScriptu se dějí neskutečné věci.

Za zmínku stojí `https://www.w3.org/TR/SVG11/animate.html`, Prezi, svgjs, D3js, `https://css-tricks.com/animate-calligraphy-with-svg/`, `https://github.com/plexus/svg-slides`, `https://github.com/Moerphy/dizzy.js`, za zmínku stojí i animejs a `https://sozi.baierouge.fr`.

Za běžných okolností si lze pdf převést na rastrové obrázky a lze s nimi na webu dělat cokoliv. Ale jak přijde na užití hypertextových odkazů, vložení videí a animací, je lepší jít jinou cestou.

Rudolf Blaško se mne ptal, jestli by dokázal svou 2D animaci z Asymptote dostat do animovaného svg. Není tedy na škodu podívat se na možnost vygenerovat animaci z TEXu. Představím vám jednu z možných cest spolupráce TEXu a JavaScriptu.

Vezmeme druhý obrázek z článku Rudolfa Blaška, animace ve 2D připravená v Asymptote. Pracovně soubor `animacka.tex`, jakože součást knihy.

```
\documentclass{standalone}
\usepackage[inline]{asymptote}
\begin{document}
\begin{asy}
real cc=1.5,u=5,v=3,rv=u/v,rm=1,rt=2*u,rp=rv-rm;int n=90;
import graph; usepackage("animate");settings.tex="lualatex";
defaultpen(.25);import animation; size(0cm,6.cm);
pair wheelpoint(real t) {return (rp*cos(t*rm/rv)+cc*cos(rp*t/rv),
    rp*sin(t*rm/rv)-cc*sin(rp*t/rv));}
guide wheel(guide g=nullpath,real a,real b,int n) {real width=(b-a)/n; for(int
    i=0;i<=n;++i){real t=a+width*i;g=g--wheelpoint(t);} return g;}
real tinterval=0*rt*pi,t1=0,t2=t1+tinterval; draw(circle((0,0),rv),olive+.75);
    real t1=8.8*pi/3; animation a; pair z1=wheelpoint(t1);dot(z1,red);real
    dt=(t2-t1)/n;
for(int i=0;i<=n;++i) {save();
  real t=t1+dt*i,kx=rp*cos(rm*t/rv),ky=rp*sin(rm*t/rv);
  filldraw(circle((kx,ky),cc),.2paleblue+white,.2paleblue+white+.5);
  draw((0,0)--(rv*cos(rm*t/rv),rv*sin(rm*t/rv)),lightblue);
```

```
  if (t>0) {filldraw((kx,ky)--arc((kx,ky),rm,180*rm*t/rv/pi,
    -180*rp*t/rv/pi)--cycle, white+.75blue+opacity(.25),drawpen=lightblue);}
  draw(circle((0,0),rv),olive+.75);label("$K$",(-.6*rv,-.75*rv),SW,olive);
  draw(circle((0,0),rp),dotted+blue+white);
  draw(circle((0,0),rp-cc),yellow+.35red);
    draw(circle((0,0),rp+cc),yellow+.35red);
  label("$x$",(rv+.25,0),N);draw((-rv-.25,0)--(rv+.25,0));
  label("$y$",(0,rv+.25),W);draw((0,-rv-.25)--(0,rv+.25));
  draw(wheel(0,10*pi,8*n),dotted+red);draw(circle((kx,ky),rm),blue+.75);
  label("$k$",(kx-.6,ky-.75),SW,blue);draw((kx,ky)--wheelpoint(t),black+.625);
  dot((kx,ky));dot(wheelpoint(t),red+black); draw(wheel(t1,t,8*max(1,i)),red+.5);
dot(wheelpoint(0),red+black);draw(wheel(0,t1,8*n),red+.5);
  label("\scriptsize$t="+string(t,7)+"$",(.3*rv,-rv),SE,blue);
  a.add();restore();}
erase(); label(a.pdf(delay=250, "buttonsize=10pt, controls, loop, palindrome",
    multipage=false));
\end{asy}
\end{document}
```

Získáme soubor `animacka-1.asy`, když si zavoláme:

```
$ lualatex animacka.tex
```

Tento soubor podsuneme Asymptote:

```
$ asy -vv animacka-1.asy
```

Získáme především soubor `_animacka-1.pdf`.

Připravíme si pomocný soubor `jadro.tex`. Ten nám pomůže s výrobou vrstveného dvi se značkami pro `dvisvgm`.

```
\documentclass[dvisvgm]{standalone}
\usepackage[palindrome, controls=all]{animate}
\usepackage{graphicx}
\begin{document}
\animategraphics{8}{_animacka-1}{}{}
\end{document}
```

Spustíme:

```
$ dvilualatex jadro.tex # nebo: lualatex --output-format=dvi jadro.tex
$ dvilualatex jadro.tex
```

Vzniká nám soubor `jadro.dvi`. Ten už převedeme do svg.

```
$ dvisvgm --exact --zoom=-1 --page=- jadro.dvi
```

Výsledné svg již můžeme otevřít, např. přes

```
$ firefox jadro.svg
$ google-chrome jadro.svg
$ chromium jadro.svg
$ opera jadro.svg
```

Pokud bychom si naopak přáli zařadit `jadro.svg` na webovou stránku, mustr by vypadal jako v tomto pracovním souboru `webovka.html`:

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html;charset=utf-8">
</head>
<body>
  <object width="400px" type="image/svg+xml" data="jadro.svg"></object>
</body>
</html>
```

4. svganimation

Je tu ještě jiná možnost. A to získat sérii nezávislých svg, jeden svg soubor vzniklý z jednoho snímku či jedné strany pdf dokumentu. To bychom u naší ukázky předchozí kapitoly získali z mnohastránkového pdf takto:

```
$ dvisvgm --pdf --exact --zoom=-1 -o "%f-%0p" --page=- _animacka-1.pdf
```

Parametr `-o` nám zajistí název souboru bez dodatečných nul. Nástrojů bychom našli nespočet, mě zaujal projekt na Syracuse:

https://melusine.eu.org/syracuse/G/svganimation

Zde je několik ukázek:

https://melusine.eu.org/syracuse/G/svganimation-exemples

První úkol je nástroj stáhnout. Lze to přes tlačítko *tree snapshot* z

https://melusine.eu.org/syracuse/G/git/?p=svganimation.git;a=tree

Pro automatizéry z příkazového řádku:

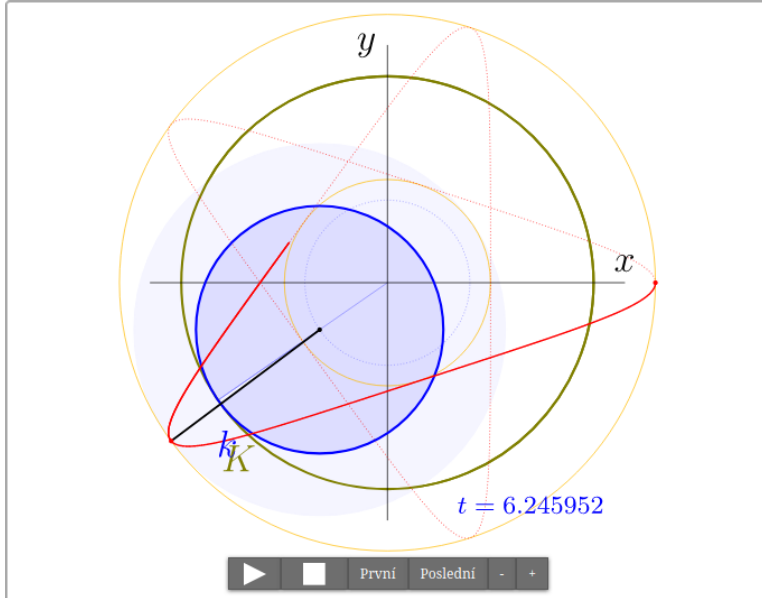
```
$ curl -o svganimation.tgz "https://melusine.eu.org/syracuse/G/git/\
  ?p=svganimation.git;a=snapshot;h=HEAD;sf=tgz"
  # nebo místo curl -o užít wget -O
$ tar xvf svganimation.tgz
$ cd svganimation-HEAD-9fed6b5/ # v mém konkrétním případě
```

Nahlédl jsem na ukázky, vytvořil potřebné složky a nakopíroval 91 souborů `_animacka-1-*.svg`.

```
$ cd ellipsographe/
$ mkdir animacka-1
$ cp animation.html animacka-1.html
$ cp <zdroj>/_animacka*.svg animacka-1/
```

V souboru `animacka-1.html` jsem upravil co bylo potřeba: popisky, otevření prvního souboru, rozsah animace, rychlost ap. Upravit popisky se dá v souboru `../SVGPlayerOne.js`. Zde je ukázka. Kvůli rozsahu zdrojový kód nepřikládám.

Asi by stačila knihovna `jinja2` v Pythonu a obecná šablona by za chvíli byla hotová.



Koho by tato oblast zajímala víc, nechť jsou takové osobě inspirací nápady na https://tex.stackexchange.com/questions/473936.

5. *Hello, world!* od balíčku **media4svg** v0.4

Jeden z posledních experimentů v TEXovém světě je balíček media4svg, který umožňuje při exportu do svg uložit audio a videostopy. Dokumentace je ještě v textové formě, nikoliv v pdf. Jedná se o jistý pokus generování snímků jako u pdf přes beamer nebo nástroj typu `powerline`.

```
$ youtube-dl -o linus.mp4 https://www.youtube.com/watch?v=CYvJPra7Ebk
$ ffmpeg -i linus.mp4 -vn linus.mp3
```

Připravíme si soubor `export-media.tex`:

```
\documentclass[dvisvgm,hypertex,aspectratio=169]{beamer}
\usefonttheme{serif}
\usepackage[utf8]{luainputenc} \usepackage[T1]{fontenc}
\usepackage[embed=false]{media4svg} \usepackage{menukeys,siunitx,calc}
\usepackage[totpages]{zref} \usepackage{atbegshi}
\usepackage{tikz} \usepgflibrary{arrows.meta}
\setbeamertemplate{navigation symbols}{}
\def\navBtnSize{9pt} \def\navBtnLnWd{1.6pt}
\AtBeginShipout{%
  \AtBeginShipoutAddToBox{%
    \special{dvisvgm:raw
```

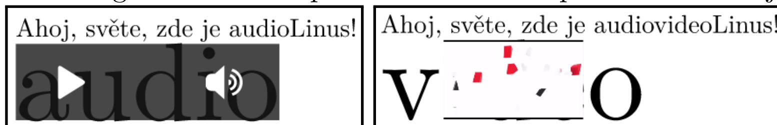
```
      <defs><script type="text/javascript">%
      <![CDATA[%        document.addEventListener('keydown',function(e){%
          if(e.key=='PageDown'){\ifnum\thepage<\ztotpages
    document.location.replace('\jobname-\the\numexpr\thepage+1\relax.svg');%
            \fi%
          }else if(e.key=='PageUp'){\ifnum\thepage>1
    document.location.replace('\jobname-\the\numexpr\thepage-1\relax.svg');%
            \fi%
        }});%
      ]]>%
      </script></defs>%
  }}%
\AtBeginShipoutUpperLeftForeground{%
  \raisebox{-\dimexpr\height+0.5ex\relax}[0pt][0pt]{\makebox[\paperwidth][r]{%
    \color{structure!40!}%
    \ifnum\thepage>1%
      \href{\jobname-\the\numexpr\thepage-1\relax.svg}{%
        \tikz{\filldraw[black!0!] (-1pt,-\dimexpr\navBtnSize/2+1pt\relax)
          rectangle
    (\dimexpr\navBtnSize+1pt\relax,\dimexpr\navBtnSize/2+1pt\relax);
          \draw[{Straight Barb[round]}-,line width=\navBtnLnWd]
    (-1pt,0)--(\navBtnSize,0);}}%
    \else%
      \textcolor{lightgray}{\tikz{\filldraw[black!0!]
    (-1pt,-\dimexpr\navBtnSize/2+1pt\relax)
        rectangle
    (\dimexpr\navBtnSize+1pt\relax,\dimexpr\navBtnSize/2+1pt\relax);
        \draw[{Straight Barb[round]}-,line width=\navBtnLnWd]
    (-1pt,0)--(\navBtnSize,0);}}%
    \fi\hspace{0.5ex}%
    \ifnum\thepage<\ztotpages%
      \href{\jobname-\the\numexpr\thepage+1\relax.svg}{%
        \tikz{\filldraw[black!0!] (-1pt,-\dimexpr\navBtnSize/2+1pt\relax)
          rectangle
    (\dimexpr\navBtnSize+1pt\relax,\dimexpr\navBtnSize/2+1pt\relax);
          \draw[-{Straight Barb[round]},line width=\navBtnLnWd]
    (-1pt,0)--(\navBtnSize,0);}}%
    \else%
      \textcolor{lightgray}{\tikz{
        \filldraw[black!0!] (-1pt,-\dimexpr\navBtnSize/2+1pt\relax)
          rectangle
    (\dimexpr\navBtnSize+1pt\relax,\dimexpr\navBtnSize/2+1pt\relax);
          \draw[-{Straight Barb[round]},line width=\navBtnLnWd]
    (-1pt,0)--(\navBtnSize,0);}}%
    \fi\hspace{0.5ex}%
```

```
}}}}%
\begin{document}
\begin{frame}{Audio}
Ahoj, světe, zde je audioLinus!\par
\includemedia[controls,width=4cm,keepaspectratio]{audio}{linus.mp3}
\end{frame}
\begin{frame}{Audiovideo}
Ahoj, světe, zde je audiovideoLinus!\par
\includemedia[controls,width=4cm,keepaspectratio]{video}{linus.mp4}
\end{frame}
\end{document}
```

Spustíme:

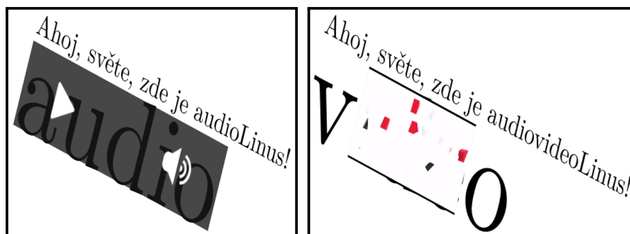
```
$ dvilualatex export-media.tex
$ dvilualatex export-media.tex
$ dvisvgm -n --bbox=papersize --font-format=woff2 --zoom=-1 --page=-
    export-media.dvi
```

Získáme dvě svg s možností si pustit zvukovou stopu a video. Zde je výřez.



Pokud třetí řádek nahradíme za následující, získáme výsledek i s geometrickou transformací.

```
$ dvisvgm -n --transform="R20,w/3,2h/5 T1cm,1cm S2,3" --page=- export-media.dvi
```



6. Náhled na interaktivitu závěrem: LaTeX2JS

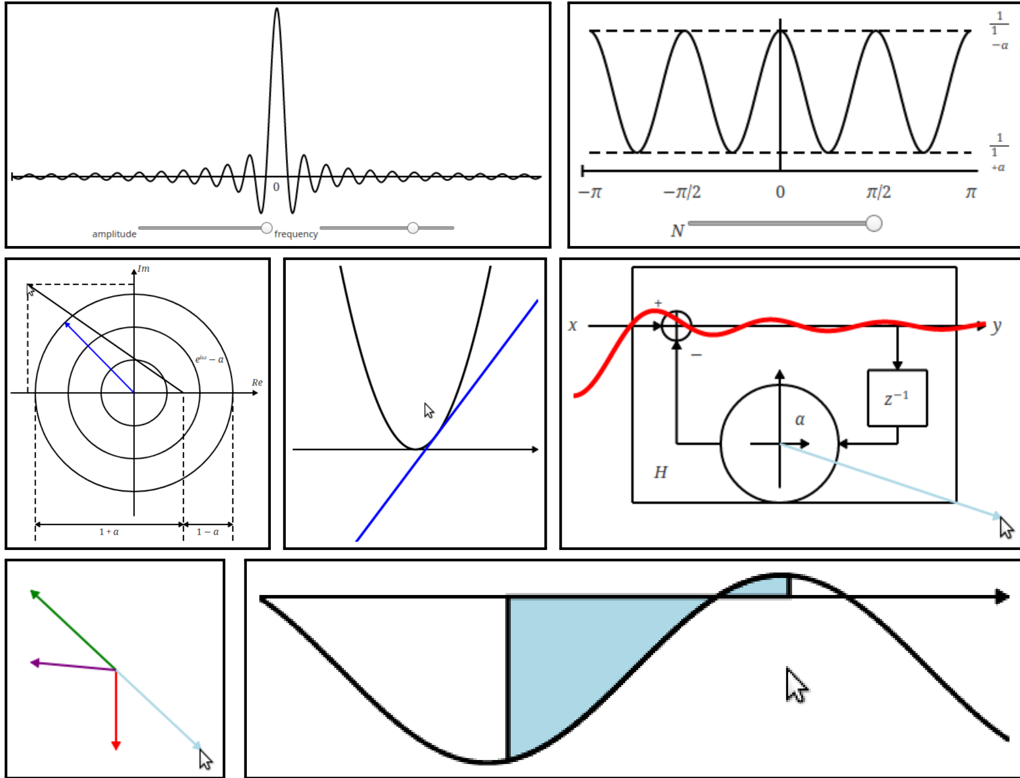
Když opomineme GeoGebru a další vhodné nástroje, zde je zajímavý experiment, na který bych rád poukázal. O projektu LaTeX2HTML5 od Dana Lynche jsem poprvé četl přes TeX.SX na stránkách `https://mathapedia.com/books/31/sections/169/400`. Ten se postupně rozšířil do obecnějšího projektu LaTeX2JS, viz `https://github.com/pyramation/LaTeX2JS`.

Autor je aktivní a některé chyby, na které jsem upozornil, upravil do několika dnů. Nyní projekt rozšiřuje vedle výstupu do HTML5 (`latex2html5`) jako

aplikaci pro React (`latex2react`) a Nuxt (`latex2vue`). To je nad rámec tohoto článku, ale je zajímavé sledovat, kam se vývoj směřuje.

Ono asi mělo dojít na nápad Petra Olšáka zmíněný na jedné konferenci TEXperience, že by měl TEX převést do C++ knihoven. Škoda, že se takový nápad a podobné pokusy (TEX-GPC, JavaTEX, PythonTEX) neuchytily, pomohlo by programátorským polyglotům v přechodech mezi TEXem a dalšími jazyky. Naopak jít do hlouby se ukazuje jako cesta budoucnosti, viz LuaTEX.

Uzavřu své pokusy náhledy z tohoto projektu. Ukázky lze interaktivně nastavit či sledují pohyb ukazatele myši ve webovém prohlížeči.



Kontaktní adresa

Ing. Pavel Stříž, Ph.D., U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,
E-mailová adresa: `pavel@striz.cz`